Office Online and SharePoint Protocols

Tom Jebo
Sr Escalation Engineer
Agenda

- WOPI Protocol Overview
- Example network flow in Message Analyzer
- FSSHTTP Protocol Overview
- Example scenarios
- Fiddler Inspector Demo
- Resources
WOPI Protocol

Office Online and Web Apps Interoperability
How Office Online Works

Browser
  Host Frame
    App Frame

SharePoint

WOPI
  Web Application Open Platform Interface

Office Online Apps Server
Web Application Open Platform Interface Protocol

- Describes the communication between the Office Web App Server (WOPI Client) and a WOPI Server (SharePoint, OneDrive, etc.)
- Defines a set of operations that enables a client to access and change files stored on a server
- REST-based protocol (HTTP or HTTPS transport)
- Open Specification available on MSDN

- Client: renders files and provides editing functionality for files stored on the server
- Server: hosts files and provides host page for navigating to client editing and viewing UI
Example flow: Viewing a file

Browser or App

Document Host

Office Online Server

Browser

WOPi Server

WOPi Client

Request view of the file

Information required to call WOPi Client

Request view of the file

Tell me about the file

Information about the file

Give me the file

The file

View of the file

WOPI Interoperability

- “Open from URL” – How do you store your Office documents?
- Develop your own WOPI host

Web Browser (or App) → Office Online Server → Microsoft Document Hosts (Office 365, OneDrive, SharePoint) → 3rd Party Document Hosts (Document Library)
Provide a link that opens Word, Excel, or PowerPoint files in a web browser.

Enter the address of the document (http, https, or \file\share)

\\jeebs5vm1\op\generate.aspx

Create Link

Open the document in a browser.

http://jeebs5vm1/op/view.aspx?src=%5C%5Cjeebs5%5Ctemp%5CTransforming%20the

Test this link

Embed the document on a web page.

<iframe src=http://jeebs5vm1/op/embed.aspx?src=%5C%5Cjeebs5%5Ctemp%5CTransforming%20the\file\share%5C\Transforming%20the\urlrc%20parameter.docx>
Transforming the urlsrc parameter

Some WOPI actions expose parameters that hosts can use to customize the behavior of the Office Online application. For example, most actions support optional query string parameters that tell Office Online what language to render the application UI in.

These parameters are exposed in the urlsrc attribute in the discovery XML. Each of these optional parameters are contained within angle brackets (< and >), and conform to the pattern `<name=PLACEHOLDER_VALUE[&]>`, where `name` is the name of the query string parameter and `PLACEHOLDER_VALUE` is a value that can be replaced by the host. By convention all placeholder values in Office Online action URIs are capitalized.
WOPI Host Basics

Step 1: Implement WOPI using [MS-WOPI]
(http://wopi.readthedocs.org can help)

Step 2: Discovery
http://<officeonlineserver>/hosting/discovery

Step 3: Build a host page
Using action URLs from step 2
REST Endpoints for WOPI server

- **http://server/.../wopi*/files/<id>** - information
  - CheckFileInfo
  - PutRelativeFile
  - Lock/Unlock
  - ExecuteCellStorageRequest/ExecuteCellStorageRelativeRequest (FSSHTTP)
  - DeleteFile, etc...

- **http://server/.../wopi*/files/<id>/contents** - contents
  - GetFile and PutFile

These are the only two that most WOPI hosts/servers will need to implement.
1. Parse and replace Placeholder values with appropriate values (or discard them)
2. Append a WOPISrc value to the URI as a query string parameter
   i.e.: http://server/.../wopi/files/(file_id)
The WOPI flow

- Browser
  - Host Frame
    - App Frame
      - WOPI Source
      - Access Token
  - WOPI Source
  - Access Token

- Host (WOPI Server)

- App (WOPI Client)

- SharePoint

- Office Online Server
Coauthoring and Multiple Editing

Office Online Server 2016:
- Lock/Unlock
- CheckFileInfo permits editing
- PutFile (frequency)

Previous versions of Office Web Apps Server:
- CheckFileInfo: SupportsCobalt
- Implement [MS-FSSHTTP*] in WOPI host

http://wopi.readthedocs.org/en/latest/scenarios/coauth.html#coauth
Coauthoring and Multiple Editing

<table>
<thead>
<tr>
<th>Application</th>
<th>Auto-save frequency</th>
<th>PutFile access token</th>
<th>Permissions check frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>Every 30 seconds if document is updated.</td>
<td>The access token of the user who made the most recent change to the document.</td>
<td>At least every 5 minutes.</td>
</tr>
<tr>
<td>Excel</td>
<td>Every 2 minutes.</td>
<td>The access token of the user who joined the editing session most recently.</td>
<td>At least every 15 minutes.</td>
</tr>
<tr>
<td>PowerPoint</td>
<td>Every 60 seconds if document is updated.</td>
<td>The access token of the user who made the most recent change to the document.</td>
<td>At least every 5 minutes.</td>
</tr>
</tbody>
</table>
Implementation Sample

Where to start if you’re lost?
https://github.com/Microsoft/Office-Online-Test-Tools-and-Documentation/tree/master/samples

C# based web handler using IHttpHandler interface
Sample host page with app frame
Simple accesstoken implementation, no real authentication
Simple file system example
Great start to understanding a WOPI host’s role
<table>
<thead>
<tr>
<th>MessageId</th>
<th>Timestamp</th>
<th>TimeElapse</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Module</th>
<th>Operation</th>
<th>Status</th>
<th>RequestURI</th>
</tr>
</thead>
<tbody>
<tr>
<td>434</td>
<td>2016-01-20 06:10:10</td>
<td>49154</td>
<td>localhost</td>
<td>http:80</td>
<td>http</td>
<td>Operation, Status: OK (200)</td>
<td>GET</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>482</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>http</td>
<td>Operation, Status: OK (200)</td>
<td>POST</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>497</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTPS</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>492</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>GET</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>511</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>POST</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>GET</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>542</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>GET</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>585</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTP</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>715</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTP</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>906</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTP</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>926</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTP</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
<tr>
<td>945</td>
<td>2016-01-20 06:10:10</td>
<td>149886</td>
<td>jeebs5vm1</td>
<td>http:80</td>
<td>jeebs5vm1</td>
<td>Operation, Status: OK (200)</td>
<td>HTTP</td>
<td>/vshub/62b632c4a0e987c8</td>
<td></td>
</tr>
</tbody>
</table>
Message Analyzer Screenshots and Demo
Message Analyzer

• Message Analyzer is a great tool to help facilitate your protocol development
• New parsers for WOPI and FSSHTTP
• Office Online leverages the existing OOXML file format and renders into a browser-friendly design
FSSHTTP Protocol
Office Online and Web Apps Efficient File Sync
MS-FSSHTTP

- A file access and locking model for document based applications

- **Key differences to DAV:**
  - Designed for high frequency incremental sync
    - Cell storage layer
  - Rich locking semantics for coauth scenarios.
    - Exclusive and shared locking
  - Support for metadata storage for co-auth scenarios.
    - Partition support “multiple streams” associated with a file.
FSSHTTP Documents

- **MS-FSSHTTP**
  - Base document
  - XML based
  - Describes how managing locks and coauthoring works
  - Describes how MS-FSSHTTPB requests are embedded

- **MS-FSSHTTPB** (*Binary Requests for File Synchronization via SOAP Protocol*)
  - Request packaging for incremental sync (cell storage requests)
  - Embedded into MS-FSSHTTP
  - Binary based not XML based (due to constructs, primitives and overheads)

- **MS-FSSHTTPPD** (*Binary Data Format for File Synchronization via SOAP*)
  - Describes how to use data model to support incremental sync for traditional, zip, any other files
  - Only required if implementing a client
Requests and File Model

URL addressable file
E.g. http://server/ffoo.doc

Main Content Partition

Metadata Partition 1

Metadata Partition 2

Lock State

MS-FSSHTTP/B/D

FSSHTTP request package

Request
Request
Request

Requests depend on the URL addressable file.
Storage/Sync Model

• The layer; cell storage layer, in MS-FSSHTTP protocol is responsible for incremental file sync.

• Instead of storing a file as an array of bytes, we store it as structured graph of objects (the “object” graph). This graph is designed to be easy for applications to use and modify, but is not optimized for storage/sync.

• The object graph is stored efficiently in another graph which is optimized for storage and sync.
Storage/Sync Layers

Optimized for Storage and Sync (low level, primitive storage construct)

Optimized for Application Use

Application Specific Schemas (to organize cell and objects)

Applications

Storage Graph

Object Graph

OneNote Graph

Generic File Stream Model

OneNote

File Stream based apps (e.g. Word/PPT/Excel etc)

MS-FSSHHTTPB

MS-FSSHHTTP(D)/MS-ONE
Storage Graph

- File is stored as Storage Graph
- Every node is identified by a GUID ID
- Two base types of nodes
  - Immutable “Data Element” node
  - Mutable “Index” node
- Data Element nodes contain majority of user data and are serialized into binary BLOBs
- Index nodes carry no data and are serialized as key/value pairs
Storage Graph

• Sync is all about send/receive changes to this graph.
  • Made easier because most nodes are immutable.

• Server implementations typically implement the graph using a table for the index node key/value pairs and a table for the immutable data element blobs.

• The immutable nature of Data Elements, which comprise the majority of the file data, are ideal for caching.
Storage Graph Sync Example

Server

Client

Index node

Data Element Node
Data="The"

Index node

Data Element Node
Data="Cat"

Index node

Data Element Node
Data="Sat"
Storage Graph Sync Example

![Diagram of storage graph sync example]

- **Server**
  - Index node
  - Data Element Node Data="The"
  - Index node
  - Data Element Node Data="Cat"
  - Index node
  - Data Element Node Data="Sat"

- **Client**
  - Index node
  - Data Element Node Data="The"
  - Index node
  - Data Element Node Data="Cat"
  - Index node
  - Data Element Node Data="Sat"
Storage Graph Sync Example

Server

Index node

Data Element Node
Data="The"

Index node

Data Element Node
Data="Cat"

Index node

Data Element Node
Data="Sat"

Index node

Data Element Node
Data="Barked"

Client

Index node

Data Element Node
Data="The"

Index node

Data Element Node
Data="Cat"

Index node

Data Element Node
Data="Sat"
Object Graph

• A file is comprised of one or more “Cells”.
• Cells contain one or more “Objects”.
  • Unit of consistency
  • State of a cell is represented by a “Revision”
• Objects contain arbitrary data and references to other objects and/or cells.
  • Unit of change
Revisions

- A set of object states that comprise the state of a cell.
- Immutable.
- Revisions can incrementally build on other revisions.
Application Schema: Office/Zip/OPC files
Application Schema: Office/Zip/OPC files
Application Schema: Office/Zip/OPC files
Application Schema: OneNote
Application Schema: All other files

Use Microsoft RDC’s FilterMax algorithm
Scenarios - On the wire Behavior
Successful File Open of a Coauthorable Document

- A client wants to open a file for co-authoring on a protocol server.
- The protocol server is named Example
- The source file to be opened is http://Example/shared%20documents/test2.xlsx

Client

Type = Coauth, CoauthRequestType = JoinCoauthoring
Type = ScemaLock, Lock type = Get Lock
Type = Cell, D/L file content or metadata

Error = Success
Error = Success
Error = Success

Server
Un-Successful File Open of a Coauthorable Document

- A client wants to open a file on a protocol server. This file is not a coauthorable document
- The protocol server is named Example
- The source file to be opened is http://Example/shared%20documents/test2.xlsx

```
Client
Type = Coauth, CoauthRequestType = JoinCoauthoring
Type = ScemaLock, Lock type = Get Lock
Type = Cell, D/L file content or metadata

Error = NumberOfCoauthorsReachedMax,
Error = 0x80004005 (E_FAIL)
Error = Success. Read only

Server
```
Successful File Open of a Document that Is Not Coauthorable

- A client wants to open a file on a protocol server. This file is not a coauthorable document.
- The protocol server is named Example.
- The source file to be opened is [http://Example/shared%20documents/test2.xlsx](http://Example/shared%20documents/test2.xlsx).

Client

Type = Exclusive Lock, Lock type = Get Lock
Type = Cell, D/L file content or metadata

Server

Error = Success
Error = Success
Un-Successful File Open of a Document that Is Not Coauthorable

- A client wants to open a file on a protocol server. This file is not a coauthorable document
- The protocol server is named Example
- The source file to be opened is http://Example/shared%20documents/test2.xlsx

The protocol server's actions:
- Type = Exclusive Lock, Lock type = Get Lock
- Type = Cell, D/L file content or metadata

Error = FileAlreadyLockedOnServer, ErrorMessage="EXAMPLE\ZZZZZ"
Error = Success. Read only mode
Office Inspectors for Fiddler

- Fiddler parsers available for FSSHTTP/B and WOPI protocols
- Github: [https://github.com/OfficeDev/Office-Inspectors-for-Fiddler](https://github.com/OfficeDev/Office-Inspectors-for-Fiddler)
- Level of detail is greater than MA Parsers for some packets
DEMO
Summary

Office Online:
- Provides the ability to work and share Office files from anywhere with an internet connection, from almost any device.
- Is an open technology
- Provides easy interoperability
- Integrate your own document library or file server with Office Online
Availability & Resources

• Office Web App Server (and Office Online) are free
  • Viewing is always free
  • Editing may have associated cost

• [MS-WOPI] is available on MSDN
  • Regular updates

• Office Online integration
  • http://wopi.readthedocs.org/en/latest/

• Support: dochelp@Microsoft.com
Thank You!

Questions?